Filter：

## Filter 函数

**注意:** 滤镜通常使用百分比 (如：75%), 当然也可以使用小数来表示 (如：0.75)。

|  |  |
| --- | --- |
| **Filter** | **描述** |
| none | 默认值，没有效果。 |
| blur(px) | 给图像设置高斯模糊。"radius"一值设定高斯函数的标准差，或者是屏幕上以多少像素融在一起， 所以值越大越模糊；  如果没有设定值，则默认是0；这个参数可设置css长度值，但不接受百分比值。 |
| brightness(%) | 给图片应用一种线性乘法，使其看起来更亮或更暗。如果值是0%，图像会全黑。值是100%，则图像无变化。其他的值对应线性乘数效果。值超过100%也是可以的，图像会比原来更亮。如果没有设定值，默认是1。 |
| contrast(%) | 调整图像的对比度。值是0%的话，图像会全黑。值是100%，图像不变。值可以超过100%，意味着会运用更低的对比。若没有设置值，默认是1。 |
| drop-shadow(h-shadow v-shadow blur spread color) | 给图像设置一个阴影效果。阴影是合成在图像下面，可以有模糊度的，可以以特定颜色画出的遮罩图的偏移版本。 函数接受<shadow>(在CSS3背景中定义)类型的值，除了"inset"关键字是不允许的。该函数与已有的box-shadow box-shadow属性很相似；不同之处在于，通过滤镜，一些浏览器为了更好的性能会提供硬件加速。<shadow>参数如下：  **<offset-x>** **<offset-y>** (必须)  这是设置阴影偏移量的两个 <length>值. **<offset-x>** 设定水平方向距离. 负值会使阴影出现在元素左边. **<offset-y>**设定垂直距离.负值会使阴影出现在元素上方。查看**<length>**可能的单位. **如果两个值都是0**, 则阴影出现在元素正后面 (如果设置了 <blur-radius> and/or <spread-radius>，会有模糊效果).  **<blur-radius>** (可选)  这是第三个code><length>值. 值越大，越模糊，则阴影会变得更大更淡.不允许负值 若未设定，默认是0 (则阴影的边界很锐利).  **<spread-radius>** (可选)  这是第四个 <length>值. 正值会使阴影扩张和变大，负值会是阴影缩小.若未设定，默认是0 (阴影会与元素一样大小).  注意: Webkit, 以及一些其他浏览器 不支持第四个长度，如果加了也不会渲染。    **<color>** (可选)  查看 <color>该值可能的关键字和标记。若未设定，颜色值基于浏览器。在Gecko (Firefox), Presto (Opera)和Trident (Internet Explorer)中， 会应用color**color**属性的值。另外, 如果颜色值省略，WebKit中阴影是透明的。 |
| grayscale(%) | 将图像转换为灰度图像。值定义转换的比例。值为100%则完全转为灰度图像，值为0%图像无变化。值在0%到100%之间，则是效果的线性乘子。若未设置，值默认是0； |
| hue-rotate(deg) | 给图像应用色相旋转。"angle"一值设定图像会被调整的色环角度值。值为0deg，则图像无变化。若值未设置，默认值是0deg。该值虽然没有最大值，超过360deg的值相当于又绕一圈。 |
| invert(%) | 反转输入图像。值定义转换的比例。100%的价值是完全反转。值为0%则图像无变化。值在0%和100%之间，则是效果的线性乘子。 若值未设置，值默认是0。 |
| opacity(%) | 转化图像的透明程度。值定义转换的比例。值为0%则是完全透明，值为100%则图像无变化。值在0%和100%之间，则是效果的线性乘子，也相当于图像样本乘以数量。 若值未设置，值默认是1。该函数与已有的opacity属性很相似，不同之处在于通过filter，一些浏览器为了提升性能会提供硬件加速。 |
| saturate(%) | 转换图像饱和度。值定义转换的比例。值为0%则是完全不饱和，值为100%则图像无变化。其他值，则是效果的线性乘子。超过100%的值是允许的，则有更高的饱和度。 若值未设置，值默认是1。 |
| sepia(%) | 将图像转换为深褐色。值定义转换的比例。值为100%则完全是深褐色的，值为0%图像无变化。值在0%到100%之间，则是效果的线性乘子。若未设置，值默认是0； |
| url() | URL函数接受一个XML文件，该文件设置了 一个SVG滤镜，且可以包含一个锚点来指定一个具体的滤镜元素。  例如：  filter: url(svg-url#element-id) |
| initial | 设置属性为默认值，可参阅： [CSS initial 关键字](http://www.runoob.com/cssref/css-initial.html) |
| inherit | 从父元素继承该属性，可参阅：[CSS inherit 关键字](http://www.runoob.com/cssref/css-inherit.html) |

transition

transition 属性是一个简写属性，用于设置四个过渡属性：

* transition-property
* transition-duration
* transition-timing-function
* transition-delay

**注释：**请始终设置 [transition-duration](http://www.w3school.com.cn/cssref/pr_transition-duration.asp" \o "CSS3 transition-duration 属性) 属性，否则时长为 0，就不会产生过渡效果。

|  |  |
| --- | --- |
| **默认值：** | all 0 ease 0 |
| **继承性：** | no |
| **版本：** | CSS3 |
| **JavaScript 语法：** | *object*.style.transition="width 2s" |

clientHeight：元素客户区的大小，指的是元素内容及其边框所占据的空间大小（经过实践取出来的大多是视口大小）

clientWidth        //返回元素的宽度（包括元素宽度、内边距，不包括边框和外边距）

offsetWidth       //返回元素的宽度（包括元素宽度、内边距和边框，不包括外边距）

offsetHeight      //返回元素的高度（包括元素高度、内边距和边框，不包括外边距）

onresize 事件会在窗口或框架被调整大小时发生。

CSS1Compat

由于历史的原因，各个浏览器在对页面的渲染上存在差异，甚至同一浏览器在不同版本中，对页面的渲染也不同。在W3C标准出台以前，浏览器在对页面的渲染上没有统一规范，产生了差异(Quirks mode或者称为Compatibility Mode)；由于W3C标准的推出，浏览器渲染页面有了统一的标准(CSScompat或称为Strict mode也有叫做Standars mode)，这就是二者最简单的区别。  
  
     W3C标准推出以后，浏览器都开始采纳新标准，但存在一个问题就是如何保证旧的网页还能继续浏览，在标准出来以前，很多页面都是根据旧的渲染方法编写的，如果用的标准来渲染，将导致页面显示异常。为保持浏览器渲染的兼容性，使以前的页面能够正常浏览，浏览器都保留了旧的渲染方法（如：微软的IE）。这样浏览器渲染上就产生了Quircks mode和Standars mode，两种渲染方法共存在一个浏览器上。  
      
     那么浏览器究竟该采用哪种模式渲染呢？这就引出的DTD，既是网页的头部声明，浏览器会通过识别DTD而采用相对应的渲染模式：  
  
1. 浏览器要使老旧的网页正常工作，但这部分网页是没有doctype声明的，所以浏览器对没有doctype声明的网页采用quirks mode解析。    
2. 对于拥有doctype声明的网页，什么浏览器采用何种模式解析，这里有一张详细列表可参考：http://hsivonen.iki.fi/doctype/    
3. 对于拥有doctype声明的网页，这里有几条简单的规则可用于判断：对于那些浏览器不能识别的doctype声明，浏览器采用strict mode解析  
4. 在doctype声明中，没有使用DTD声明或者使用HTML4以下（不包括HTML4）的DTD声明时，基本所有的浏览器都是使用quirks mode呈现，其他的则使用strict mode解析。  
5. 可以这么说，在现有有doctype声明的网页，绝大多数是采用strict mode进行解析的。  
6. 在ie6中，如果在doctype声明前有一个xml声明(比如:<?xml version="1.0" encoding="iso-8859-1"?>)，则采用quirks mode解析。这条规则在ie7中已经移除了。  
  
  
     Quirks mode和Standars mode最大的不同就是提现在对盒模式的解释上，这也是我们在js里要注意的地方。  
     什么是盒模式？ 这是针对块级元素说的，说白了就是把块级元素想像成一个装东西的盒子，而margin,padding,border,width这些css属性构成了盒模式。    
  
在Standars mode中：   
  
元素真正的宽度 = margin-left   +   border-left-width   +   padding-left   + width   +   padding-right   +   border-right-width   +   margin-right;    
  
在Quirks mode中：   
  
width则是元素的实际宽度，内容宽度 = width   -   (margin-left   +   margin-right   +   padding-left   +   padding-right   +   border-left-width   +   border-right-width)  
  
1. 内联元素 ，例如<a>|<span>等，定义上下边界不会影响到行高（line-height），内联元素距离上一行元素的距离由行高决定，而不是填充或边界。    
2.   内联元素（display: inline） 内联元素不需要在新行内显示，而且也不强迫其后的元素换行，如a|em|span等都为内联元素。内联元素可以为任何其他元素的子元素。  
3. 浮动元素（无论左或者右浮动）边界不压缩 ，且若浮动元素不声明宽度，则其宽度趋向于0，即压缩到其内容能承受的最小宽度。    
4. 如果盒中没有内容，则即使定义了宽度和高度都为100%，实际上只占0% ，因此不会被显示，此点在采取层布局的时候需特别注意。    
5. 边界值可为负，其显示效果各浏览器可能不相同。    
6. 填充值不可为负。    
7. 边框默认的样式（border-style）为不显示（none）。  
  
****在js中如何判断当前浏览器正在以何种方式解析？****     
document对象有个属性compatMode ,它有两个值：  
  
BackCompat     对应quirks mode  
CSS1Compat     对应strict mode